TO: Verne

COMPUTER ENGINEERING

A DEC VIEW OF HARDWARE SYSTEMS DESIGN

C. GORDON BELL · J. CRAIG MUDGE · JOHN E. McNAMARA

DIGITAL PRESS

TO: Jennie

CONTENTS

TO: Jennie

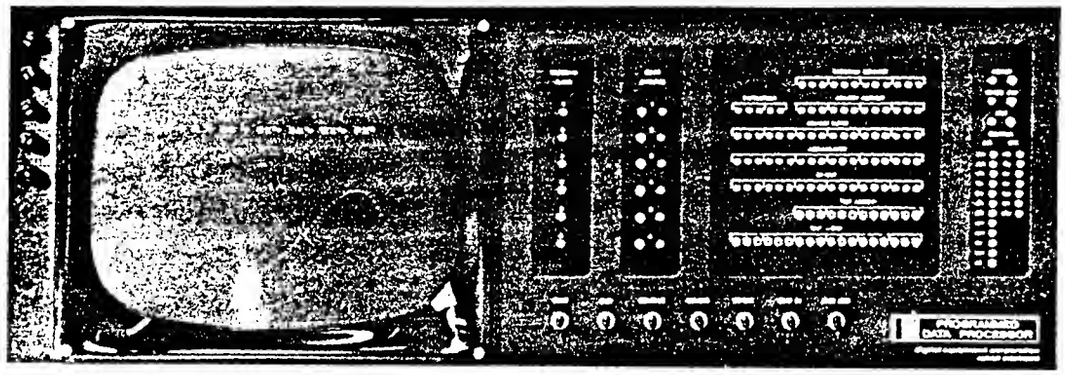

Figure 9. PDP-1/A CRT console.

Figure 10. PDP-1/B at BBN (circa 1960).

but this design was subsequently dropped for cost reasons. The use of a cathode ray tube integrated into the console never returned to the DEC main line of computers, except briefly in a few PDP-6s and in the LINC and PDP-12 laboratory computers. In modern fourth generation (large-scale integration) computers, the entire computer is integrated into the cathode ray tube housing.

Bolt, Beranek, and Newman (BBN), a consulting firm in Cambridge, Massachusetts, purchased the first production machine (1/B) for delivery in November 1960. This machine is shown in Figure 10. A third machine, similar to

the 1/
use.

Afte
clear t
prove
and in
require
tween t
liable.
the PD
console
in Figu
DEC a
prove a
of the
The P
mounti
PDP-4
mained
directio
used in
minicon

Aside from the experience gained from having to produce computers that could run unattended and without service, the most important result of the ITT order was that it allowed DEC to build a number of identical machines without special engineering. This in turn provided a production base with decreased costs (as described in Chapter 3) and a discipline to be less special systems oriented. The first few machines ordered by other customers had been nearly all different, requiring DEC to build options that were sold only a few times. In addition, many of those machines had interfaces that were unique to the applications.

It should be noted that because the hardware for the PDP-1 was relatively inexpensive, DEC could afford to stock an ample supply of basic modules for building special interfaces. Constructing interfaces and specialized hardware was relatively easy compared to modern day hardware design. Also, design errors could be corrected with simple wiring changes – a much easier process than that demanded by the modern day, where expensive printed circuit boards have fine etch lines to be cut and read-only memories to be changed. Finally, the special interfaces and controllers for the PDP-1 were quite simple compared to modern designs.

While the ITT sale was important to DEC's future, the Bolt, Beranek, and Newman (BBN) sale was important to the future of the entire computer industry because it was one of the events leading to the development of timesharing. A number of computer scientists at M.I.T. and BBN believed that it was necessary to provide interactive access to computers. The only way to make this economically viable was to simultaneously share the computer among the users. Three experiments were carried out to demonstrate its feasibility: the IBM 7090 system at M.I.T. [Corbato *et al.*, 1962] which later became the Compatible Time Sharing System (CTSS), the multiuser PDP-1 at M.I.T. [Dennis, 1964] which was operational in 1963, and the shared PDP-1 at BBN [McCarthy *et al.*, 1963].

Batch multiprogramming [Strachey, 1959] was an important part of the design of the Stretch computer [Buchholz, 1962] and the Atlas computer [Kilburn *et al.*, 1962]. They were oriented toward hardware efficiency in that they aimed for high utilization of all components. Timesharing, on the other hand, was concerned with the efficiency of the people trying to use the computer – the efficiency of the man-computer interaction [Corbato *et al.*, 1962].

A set of requirements was identified for a timesharing system. Unless the workload was restricted to programs that were specially designed to run concurrently and to programs that were error-free, one needed the following:

1. Memory protection.
2. Program and data relocatability.
3. A supervisor program.
4. A timed return to the supervisor.
5. Interpretive execution of the I/O instructions.

The BBN timesharing system began operation in September 1962. Five teleprinter users shared the upper 4 Kwords of memory; the lower 4 Kwords held the supervisor program, called the "channel 17 routine." The modifications to the PDP-1 to effect timesharing were embodied in the "restricted mode" of operation. They matched the above requirements in the following way:

1. **Memory protection.** Switching between the two 4-Kword areas required the use of an I/O instruction.
2. **Program and data relocatability.** Because only one user was resident at one time, this was not needed.
3. **A supervisor program.** The channel 17 clock routine fulfilled this function.
4. **A timed return to the supervisor.** The channel 17 clock generated an interrupt every 20 milliseconds.

5. **Interpretive execution of I/O instructions.** Whenever the PDP-1 was in restricted mode, an attempt to obey an I/O instruction caused a sequence break.

The TYC Control Language, a debugging aid adapted from the DDT language devised for the PDP-1 and its predecessor languages, was regarded as important because it allowed direct language program debugging. The "restricted mode" modifications, a high speed swapping drum, and the use of the new multiport memory designed for the PDP-6 formed the PDP-1/D design. Timeshared computers were built and operated at BBN, Stanford, and M.I.T. These timesharing efforts later influenced the use of timesharing in the PDP-6 (Chapter 21).

THE PDP-4

About two years after the PDP-1 was first shown, the notion of a much smaller machine developed during discussions of process control applications with Foxboro Corporation and various other customers. A machine called the DC-12 Digital Controller was proposed. This would be a 12-bit computer oriented toward process control data collection and laboratory data processing. During the preparation of the proposal, the CDC 160 was studied, and the DEC engineers briefly considered building a copy or version of the 10-bit L-1 computer designed by Wes Clark at Lincoln Laboratory. However, the principal idea input for the Digital Controller came from another Wes Clark computer, the Laboratory Instrument Computer (LINC).

The DC-12 Digital Controller was never built by that name; instead, it became the PDP-5 (Chapter 7). Some of the ideas studied in the LINC and L-1 were used in other DEC machines, including the machine that became the PDP-1 successor, the PDP-4 (Figure 14). The PDP-2 designation was saved for a possible 24-bit machine, but none was ever built. DEC also never built a PDP-3, although one was designed on paper as a 36-bit machine.*

The decision to make the next machine an 18-bit machine, rather than a 12-bit machine, was taken very lightly when it was made in December of 1962. In retrospect, it may have been a poor decision, but the reasoning went somewhat as follows.

Based on the programming experience of the TX-0, Gordon Bell felt that an 18-bit machine significantly simpler than the PDP-1 could be built and that simple machines with few instructions for a given number of data-types would perform nearly as well as those with more instructions. This feeling was based on the use of Whirlwind, TX-0 as it evolved through its various versions, and the PDP-1. This was later proven to be true, as the PDP-4 was implemented in less than half the space of the PDP-1 and provided 5/8 the performance for 1/2 the price. There is some question, however, as to how much of the size reduction was due to the simpler architecture, how much to the substantially better logic design implementation, and how much to the increased logic packing density.

Gordon Bell had conceived the idea of auto-incrementing memory registers. This allowed vectors to be accessed easily instead of using index registers. The auto-incremented memory registers performed about as well as index registers and were much less expensive to implement.

The PDP-1 had used one's complement arithmetic, which was especially poor for the fast multiple precision operations and floating-point arithmetic that DEC's customers needed.

*In 1960 a customer (Scientific Engineering Institute, Waltham, Massachusetts) built a PDP-3. It was later dismantled and given to M.I.T.; as of 1974, it was up and running in Oregon.

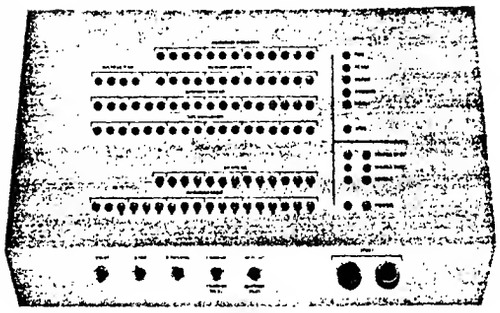

Figure 19. PDP-4 operator console.

on the console. This simplified testing by permitting easy use of an oscilloscope. In addition, simple checks on memory could be performed by using the console Read and Write switches and observing the results on the console lights.

Because the PDP-1 had been generally used in dedicated applications, the users had written their own programs. M.I.T., for example, had contributed a good macroassembler, linking loader, and interactive debugging program - DDT. BBN had contributed various subprograms. DEC had invested very little in PDP-1 software and thus had no concern for the cost of writing system software or for the concept that a new machine should capitalize on previous systems programming. It was easy for people at DEC to believe that a small part of the savings achieved by building a simpler machine could be used to pay for the writing of new software for that machine.

In the present day, designers of new computers realize that program compatibility is a constraint and that any new machine must be on an improving cost/performance line. (This is discussed in greater detail in Chapters 2 and 15.) At the time that compatibility decisions were being made with regard to the PDP-4, about 20 PDP-1s had been installed out of an eventual population of 50. Looking back from today's vantage point, a compatible machine might have been built that would have interpreted most of the PDP-1 programs and offered the same improved cost/performance ratios as the PDP-4 did, but still not have been very much larger than the original PDP-4.

The PDP-4 was a limited success. While it met the corporate profit standard, it did not sell as well as had been expected. The market demands were not as completely elastic as they had been for the PDP-1, and 5/8 of the performance for 1/2 the price was not good enough. According to the evolution model discussed in the final section of this chapter, a machine with a lower price should have had the same performance as the PDP-1, or else it should have been priced much less than the PDP-1 to compensate for the relatively poor performance. In summary, the PDP-4 was not aggressive enough in performance or in price. There is an additional reason for the poor financial showing of the PDP-4. Experience with other machines that were the first of a series, such as the PDP-5, PDP-6, LINC-8, PDP-14, and PDP-11/20, indicates that the financial performance of the first machine is always the poorest of the series, largely because of the lack of a software and hardware option base. The PDP-7, 9, 9/L, and 15 were necessary successors that used the software and hardware option base created by the PDP-4.

THE PDP-7

In many ways the original concept of the PDP-7 (or what was finally named the PDP-7) started with the design of the PDP-1/D. The initial plans were to simply repackage the PDP-1, using some higher density systems modules, and to reduce the processor cycle time. The goal was to use these changes to produce a lower price machine with much better performance. This goal was met quite well in the PDP-7, as it had a greater performance/price gain over its predecessors than any other DEC 18-bit computer.

The plan to simply repackage the PDP-1 was abandoned when consideration was given to the

21

The Evolution of the DECsystem-10

C. GORDON BELL, ALAN KOTOK,
THOMAS N. HASTINGS, and RICHARD HILL

INTRODUCTION

The project from which the PDP-6, DECsystem-10, and DECSYSTEM-20 series of scientific, timeshared computers evolved began in the spring of 1963 and continued with the delivery of a PDP-6 in the summer of 1964. Initially, the PDP-6 was designed to extend DEC's line of 18-bit computers by providing more performance at increased price. Although the PDP-6 was not designed to be a member in a family of compatible computers, the series evolved into five basic designs (PDP-6, KA10, KI10, KL10, and KL20) with over 700 systems installed by January 1978. During the initial design period, we neither understood the notions and need for compatibility nor did we have adequate technology to undertake such a task. Each successive implementation in the series has generally offered increased performance for only slightly increased cost. The KL10 and KL20 systems span a five to one price range.

TOPS-10, the major user software interface, developed from a 6-Kword monitor for the PDP-6. A second user interface, TOPS-20, introduced in 1976 with upgraded facilities, is based on multiprocess operating systems advances.

This paper is divided into seven sections. Section 2 provides a brief historical setting followed by a discussion of the initial project goals, constraints, and basic design decisions. The instruction set and system organization are given in Sections 4 and 5, respectively. Section 6 discusses the operating system, while Section 7 presents the technological influences on the designs. Sections 4 through 7 begin with a presentation of the goals and constraints, proceed to the basic PDP-6 design, and conclude with the evolution (and current state). We try to answer the often-asked questions, "Why did you do . . .?", by giving the contextual environment. Figure 1 helps summarize this context in the form of a timeline that depicts the various hardware/software technologies (above line) and when they were applied (below line) to the DECsystem-10.

HISTORICAL SETTING

The PDP-6 was designed for both a timeshared computational environment and real-time laboratory use with straightforward interfacing capability. At the initiation of the project, three timeshared computers were

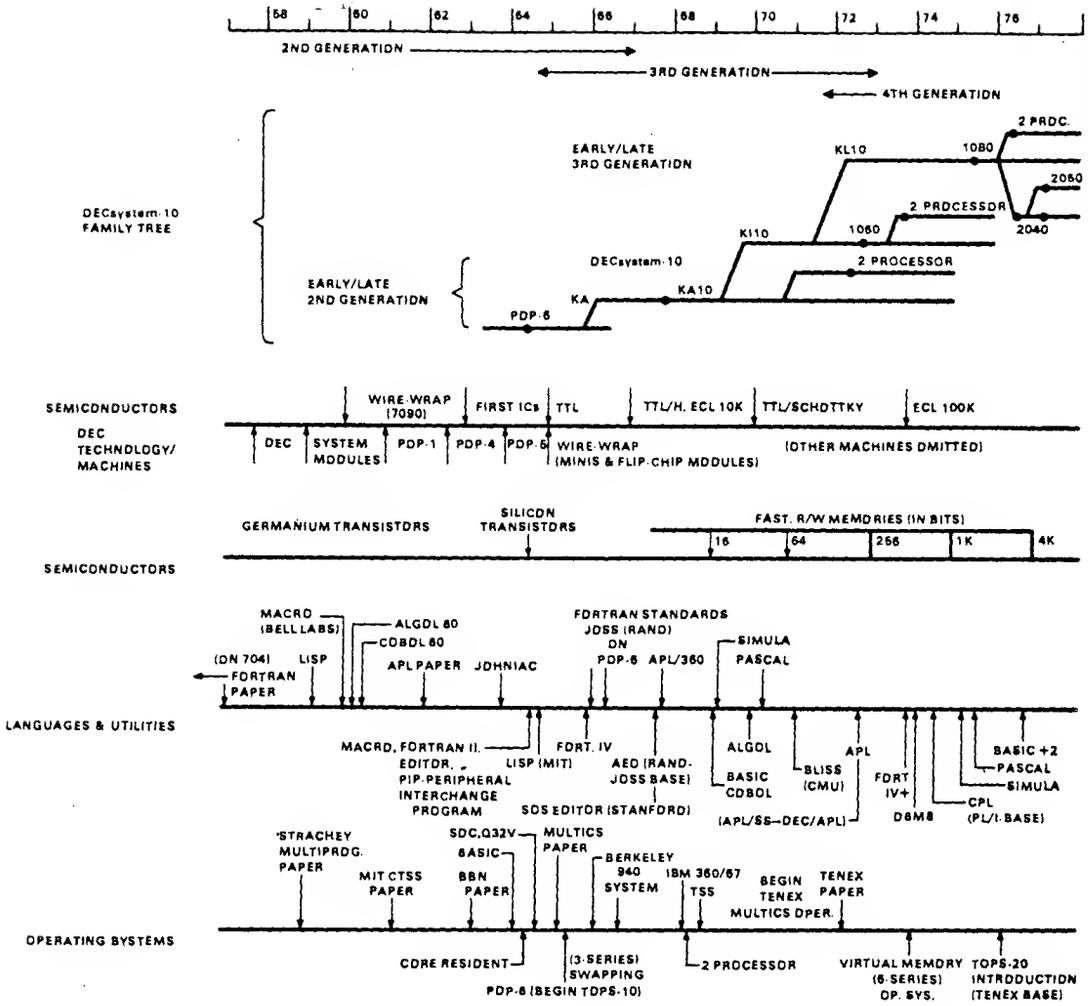

Figure 1. Timeline of DECsystem-10 evolution.

operational: a PDP-1 at Bolt, Beranek, and Newman (BBN) which used a high-speed drum that could swap a 4 Kword core image in one 34 ms revolution; an IBM 7090 system at MIT called CTSS, which provided each of 32 users a 32 Kword environment; and an AN/FSQ-32V at SDC, which could serve 40 simultaneous users.

The Bell Laboratory's IBM 7094 Operating System was a model operating system for batch users. Burroughs had implemented a multiprogrammed system on the B5000. Dartmouth was considering the design of a single language, timesharing system which subsequently became BASIC. The MIT Multics system, the Berkeley SDS 940, the Stanford PDP-1 based timeshared

system for com
BBN Tenex sys
the DECsystem

In architectu
Newell, 1971:C
cause it was a
build, but bec
good design
programmed w
tween the user
large address s
of extra codes
instruction set.
could not affo
small memory.
in use on their
avoid since ou
were generally s
could outperfor
currency and I
simple interrup

The DEC pr
in Table 1. Sale
was felt that co
$20,000 to $300
the problems
enough address
IBM machines
ties.

On the softw
DEC had been
Kword) users,
programmed
size of 4 Kwor
constraint. The
of operating sy
ther academia
ect was just be
project did not
were no peopl
users within th
signers were co
in the Cambrid
on the design (e

Utilities and languages have taken advantage of the interactive, terminal-oriented environment. Thus, highly interactive editing/debugging facilities have evolved in terms of the program's own symbols. The file/data transfer utility, PIP (for Peripheral Interchange Program) is still in existence today, although in a much enhanced form. It has since been expanded to support the peripheral devices and the data formats encountered in the DECsystem-10 memory and I/O devices. Such a utility eliminated the need for a "library" of utilities and conversion programs to transfer data between devices. Such tasks as a card-to-disk, card-to-tape, tape-to-disk, etc., conversion are controlled by a terminal using common PIP commands. PIP evolved in a somewhat *ad hoc* fashion from a 1 Kword or 2 Kword size in 1965 to 10 Kwords with substantial generality.

A powerful and sophisticated text editor, TECO (Text Editor and Corrector) was initially implemented at MIT using a graphics display. TECO is character-string oriented and requires a minimal number of keystrokes to execute commands. It included the ability to define programs to do general string substitution. As the sophistication of users was later perceived to decline, the powerful editor created training and use problems. Thus, a family of line- and character-oriented editors evolved which was easier to learn and remember. These were based on other line-oriented editors, but especially Stanford's SOS, which replaced the initial DECline editor in 1970.

Many of the higher level languages were initially produced by non-DEC groups and made available through the DEC User Society (DECUS). For example, APL, BASIC, DBMS, and IQL (an interactive query language) were purchased from outside sources and are now standard, supported products.

BLISS (Basic Language for Implementing System Software), developed at Carnegie-Mellon University, became DEC's systems programming language [Wulf *et al.*, 1971b]. A cross-compiler was subsequently developed for the PDP-11. Its use as a systems programming language has been due to the close coupling it provides to the machine, its general syntactic and block structures, and its high-quality code generator. BLISS has been used for various diagnostic programs, the BLISS Compilers, the PDP-10 APL Interpreter, recent FORTRAN-IV compilers for both PDP-10 and PDP-11, and the BASIC PLUS TWO system. BLISS has also been used extensively within DEC for computer-aided design programs.

Tenex and the TOPS-20 Operating System

Bolt, Beranek, and Newman started a project in 1969 to build an advanced operating system called Tenex which was based on a modified KA10 (including rather elaborate paging hardware). This work was influenced by both the Berkeley SDS 940 and the MIT Multics systems. Subsequently, Tenex influenced and improved the KI10 design which became the base of TOPS-20. The system was described by Bobrow *et al.* [1972], and the three major goals stated in the reference were:

I. State-of-the-Art Virtual Machine
   a. Paged virtual address space equal to or greater than the addressing capability of the processor with full provision for protection and sharing.
   b. Multiple process capability in virtual machine with appropriate communication facilities.
   c. File system integrated into virtual address space, built on multilevel symbolic directory structure with protection, and providing consistent access to all external I/O devices and data streams.
   d. Extended instruction repertoire making available many common operations as single instructions.

II. Good Human Engineering Throughout Systems

a. An executive command language interpreter which provides direct access to a large variety of small, commonly used system functions, and access to and control over all other subsystems and user programs. Command language forms should be extremely versatile, adapting to the skill and experience of the user.

b. Terminal interface design should facilitate intimate interaction between program and user, provide extensive interrupt capability, and full ASCII character set.

c. Virtual machine functions should provide all necessary options, with reasonable default values simplifying common cases, and require no system-created objects to be placed in the user address space.

d. The system should encourage and facilitate cooperation among users as well as provide protection against undesired interaction.

III. The System must be Implementable, Maintainable, and Modifiable

a. Software must be modular with well defined interfaces and with provision for adding or changing modules clearly considered.

b. Software must be debuggable and reliable, allowing use of available debugging aids and including internal redundancy checks.

c. System should run efficiently, allow dynamic manual adjustment of service if desired, and allow extensive reconfiguration without reassembly.

d. System should contain instrumentation to clearly indicate performance.

Dan Murphy (one of Tenex's designers/implementers) came to DEC and led the architecture and development effort that produced TOPS-20. The effort at DEC has been to increase the performance of TOPS-20 to be competitive with the highly tuned Monitor while not losing its generality. The TOPS-20 structure does provide increased reliability and modifiability.

HARDWARE IMPLEMENTATION

While logic and memory technology are often considered the prime determinant of the performance and cost of a computer system, fabrication and packaging technology are equally important. This section surveys logic, manufacturing, and packaging technology as it affected the various DECsystem-10 models. Table 7 summarizes those various logic and packaging technologies.

Logic

The PDP-6 used a set of logic modules that evolved from the earlier PDP-1, which in turn were derived from the Lincoln Laboratory circuits developed for the TX-0 [Mitchell, Olsen, 1956] and TX-2 [Clark, 1957] (Chapter 4) computers as part of the air defense program. These circuits were direct-coupled transistor logic and included both series and parallel transistor circuits to give great flexibility in designs. The PDP-1 circuits operated at a 5 MHz clock, and new transistors enabled the PDP-6 circuits to operate at 10 MHz. The computer's clock was based on a delay line which carried pulses generated by a pulse amplifier using pulse transformers (this too came from Lincoln Laboratory via the early work at MIT on radar and pulse transformers) The pulses were used for register transfer operations (i.e., moving data among the registers) and some logic gating.